Proceedings, 10th IEEE International Workshop on Software Specification and Design, San Diego, California, 5-7 November 2000. IEEE Computer Society, pp 105-114.

Design and Development Assessment

Steven L. Cornford, Martin S. Feather,
John C. Kelly,
Timothy W. Larson, Burton Sigal,
Jet Propulsion Laboratory,
California Institute of Technology
4800 Oak Grove Drive
Pasadena, CA 91109, USA
Steven.L.Cornford@Jpl.Nasa.Gov
Martin.S.Feather@Jpl.Nasa.Gov
John.C.Kelly@Jpl.Nasa.Gov
Timothy.W.Larson@Jpl.Nasa.Gov
Burton.Sigal@Jpl.Nasa.Gov

James D. Kiper
Department of Systems Analysis
Miami University
Oxford, OH 45056
kiperjd@muohio.edu

ABSTRACT

An assessment methodology is described and illustrated. This methodology separates assessment into the following phases (1) Elicitation of requirements; (2) Elicitation of failure modes and their impact (risk of loss of requirements); (3) Elicitation of failure mode mitigations and their effectiveness (degree of reduction of failure modes); (4) Calculation of outstanding risk taking the mitigations into account.

This methodology, with accompanying tool support, has been applied to assist in planning the engineering development of advanced technologies. Design assessment featured prominently in these applications. The overall approach is also applicable to development assessment (of the development process to be followed to implement the design).

Both design and development assessments are demonstrated on hypothetical scenarios based on the workshop's TRMCS case study. TRMCS information has been entered into the assessment support tool, and serves as illustration throughout.

Keywords

Assessment, Requirements Elicitation, Risk Management, Software Processes, Quality Assurance, Capability Maturity Model, Tradeoffs

1. INTRODUCTION

In complex and critical systems, assessments are a means to determine adequacy of designs to meet their requirements, and the adequacy of development plans to satisfactorily implement designs.

This paper outlines a methodology to performing detailed and quantitative assessments of system designs and of software development plans. The key components to this methodology are the notions of *Requirements* (what it is that the system is supposed to achieve), *Failure Modes* (things that, should they occur, will lead to loss of requirements) and *Mitigations* (design components, activities, etc., that reduce the risk of requirements loss incurred by Failure Modes). The methodology advocates the disciplined approach to elicitation of each of these, culminating in the calculation of outstanding risk taking the mitigations into account.

This approach to assessment is based upon a broader methodology for spacecraft mission assurance and planning, called Defect Detection and Prevention (DDP) [Cornford, 1998]. A computerized tool supports the real-time application of DDP. The DDP tool represents the elicited information, computes derived information (e.g., aggregate risk), and graphically displays information. The DDP tool is designed to offer modest capabilities in all these areas. It emphasizes tight coordination between its various capabilities, which accounts for its capacity to enable users to work effectively within a large space of information, discussed further in [Feather et al, 2000].

The rest of this paper is organized as follows:

The major phases of design assessment are covered first: requirements elicitation (Section 2), failure modes elicitation (Section 3), mitigations elicitation (Section 4), and assessment calculation (Section 5). For each, the methodology and tool support is described and illustrated on hypothetical scenarios within the TRMCS domain. Since the authors are by no means experts in this domain, it should be understood that the purpose of these scenarios is to illustrate the potential of the assessment methodology. Development assessment is considered next (Section 6). Conclusions follow (Section 8), and finally some further illustrations of development assessment in the TRMCS application are in an appendix.

2. REQUIREMENTS ELICITATION

Requirements elicitation is the first step to performing an assessment. The system's design will be measured against those requirements.

Requirements Elicitation - Methodology

The assessment process must establish the system's requirements, and their relative importance. All the key stakeholders must contribute to this activity, in order that no critical requirement is accidentally overlooked. Since not all requirements will be equally important, they must be weighted relative to one another. This will likely need the simultaneous involvement of experts from multiple disciplines. It is important that this establishing of the relative importance of the requirements not be biased by knowledge of the ease or difficulty of the achievement within a given design or approach.

Requirements elicitation is performed in a session at which all the stakeholders attend. A moderator directs the flow of conversation, encourages input from all stakeholders, etc. The DDP tool is used to capture the elicited requirements and display them for all attendees to see.

The stage at which the assessment takes place bounds the level of detail to which requirements can be elicited. For example, only after a detailed design has been formulated can requirements of the design's subcomponents be determined. Furthermore, it is only necessary to elicit enough detail to be able to conduct the assessment. As a result, modest capabilities for representing requirements suffice. These are discussed next.

Requirements Elicitation – Tool Support

The DDP tool offers the following capabilities for representing and manipulating requirements:

- A pre-determined set of useful attributes for requirements – e.g., title, reference (the author/source of the requirement), description (unbounded text field for length comments), and relative weight. The process (and tool) make many of the attributes optional, so that the users can make the choice of when and how much detail to provide.

- Ability to add/edit/remove requirements on the fly. It is also possible to turn "on" and "off" individual requirements.

- Tree-structured organization of requirements, permitting on-the-fly reorganizations during the elicitation process. This form of hierarchical grouping is particularly useful as the number of requirements grows.

- Bottom-up or top-down computation of requirement weights. In bottom-up computation, the stakeholders assign relative weights to the "leaf" requirements, which are aggregated upwards through the requirements tree to determine the relative weights of the parent requirements. Alternately, in top-down computation, the weight assigned to the topmost ("root") node of a requirements tree is distributed to its child nodes in proportion to their relative weights, continuing this process recursively down through the tree.

- A choice of styles for automatic numbering of requirements. Tree-structured numbering (1, 1.1, 1.2, …2, …) is the most popular.

Figure 1 shows a snapshot of the tool, displaying the requirements taken from the TRMCS case study documentation.

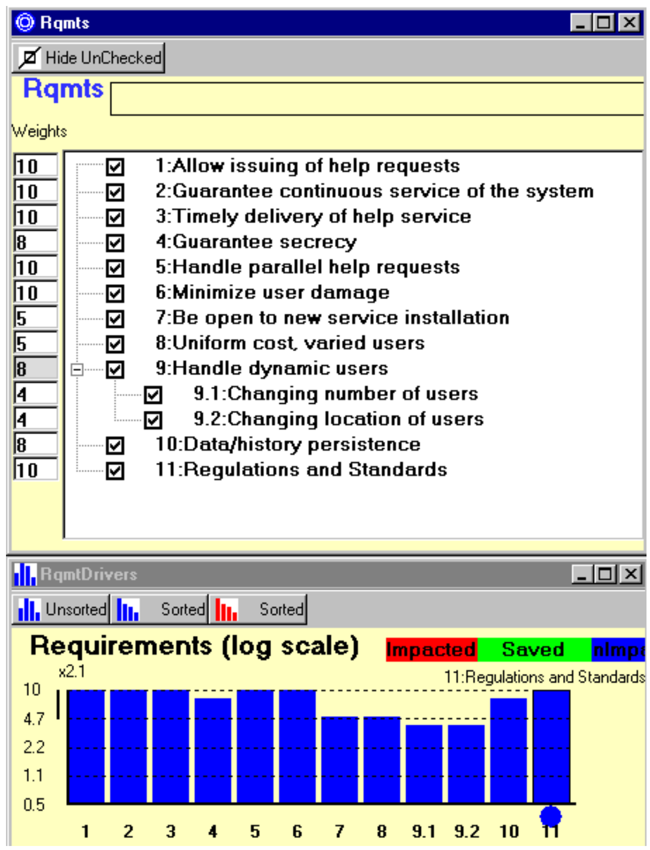

Figure 1 - Requirements and a chart of their weights

For the purposes of illustration, the case study's "Handle dynamic changes to the number and location of users" has been turned into a small tree whose parent node is “Handle dynamic users”, and whose children are “Changing number of users” and “Changing location of users”.

Requirement weights are shown pictorially in the bar chart, and the stakeholders’ assigned weights are shown in the boxes to the left of the tree. The effect of bottom-up computation of requirements is discernable in the weight of requirement 9. It’s weight, 8, is the sum of the weights assigned to its two children, and its background is automatically shaded to indicate that it is calculated, and therefore not directly editable.

3. FAILURE MODES ELICITATION

The second major step of the assessment process is the elicitation of failure modes – all the things that, should they occur, will lead to loss of requirements. This step also includes the determination of how much each failure mode impacts each requirement. For example, a power outage at the TRMCS center would adversely impact the “Guarantee of continuous service” requirement (and others) if nothing were done to compensate for it.

Failure Modes Elicitation - Methodology

As was the case for requirements, all the stakeholders should contribute to the activity of eliciting failure modes, in order that no critical failure mode is overlooked. However, determination of how much each failure mode impacts each requirement need not necessarily involve all the stakeholders simultaneously. Instead, it is typical that failure modes can be subdivided into major disciplines, and, for a given discipline, only the experts in that discipline need be involved in determining the impacts of its failure modes.

Failure modes include both external events (e.g., lightning strikes, power failures) and internal events (e.g., failure caused by a bug in the system’s software). This phase of the assessment process determines the likelihood and impact of failure modes *as if nothing were done to inhibit their occurrence or reduce their impact*. Mitigation of failure modes, by good design choices and by following good design methodologies, will be taken into account in subsequent stages of the assessment process.

We have postulated 11 major failure modes – see Figure 2. Some are consequences of external events, for example number 1, “Power outage at center.” Some may be caused by events internal to the system, for example if the design includes its own communication system over which the TRMCS system will operate, then its own failure would cause number 9, “Communications system down”. Some may be combinations of both, for example if the TRMCS deploys its own monitors that communicate using an existing paging network, then in concert these may lead to number 9, “Rudimentary connectivity from/to user.”

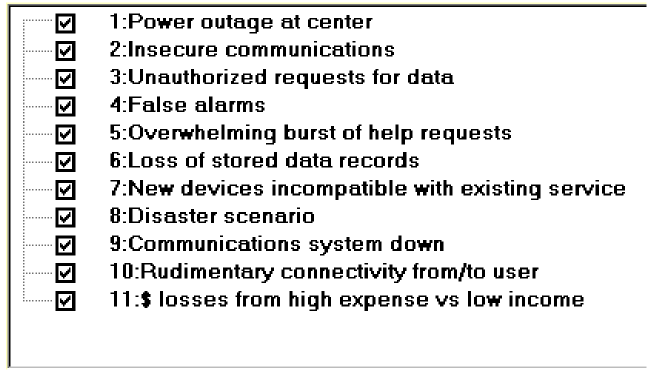

Figure 2 – hypothesized Failure Modes

Failure Modes Elicitation – Tool Support

The DDP tool’s support for representation and elicitation of failure modes is similar to that for requirements. Failure Modes have many of the same attributes; they can be organized into trees, etc. A Failure Mode does not have a weight (an attribute specific to requirements), but does have an a-priori likelihood (an attribute specific to Failure Modes).

A Failure Mode may have a different impact on different requirements. Thus impact is not an attribute of a Failure Mode alone, but is an attribute of a Failure Mode x Requirement pair. The DDP tool uses a matrix as the primary means to allow the entering/editing/inspecting of impacts. The rows of this matrix are Requirements, and the columns Failure Modes. Each inner cell holds the impact value of the cell’s column’s Failure Mode on the cell’s row’s Requirement. An impact value is a number in the range 0 to 1, where 0 corresponds to no impact whatsoever, and 1 corresponds to complete loss of the Requirement should the Failure Mode occur. An empty cell is equivalent to an entry of 0.

Figure 3 shows some hypothesized impact values for the previously listed Failure Modes on the TRMCS requirements. For example, the first row and column (shown highlighted) correspond to the Requirement “Allow issuing of help requests” and Failure Mode “Power outage at center”. The inner cell holds the value 1, indicating that a power failure will lead to complete loss of ability to issue help request. This is plausible, since the system at the center would presumably be rendered inoperable by the power failure if nothing were done to mitigate this.

The tool automatically calculates some aggregate values for impacts. These are shown in the second row from the top, and third column from the left:

- The row of aggregate values displays, for each Failure Mode, the total expected risk of that Failure Mode. For Failure Mode FM, this is computed as:

  A-priori-impact(FM) = Likelihood(FM) * ($\sum$ (R $\in$ Requirements): Weight(R) * Impact(FM,R))

RxFM

Col = Power outage at center

Row = Allow issuing of help requests

		FMs	Power	Insecu	Unauth	False	Overwl	Loss	New	Disast	Commu	Rudim	$
Rqmts	Rqmts	Totals	51.8	8	11	14	25	10.2	10	45	20	18.8	4.5
Allow		41	1				0.2		0.5	0.7	1	0.7	
Guarar		28	1							0.8	1		
Timely		35	1			0.8	0.7	0.1		0.9			
Guarar		16		1	1								
Handle		31	1			0.3	0.8			1			
Minimi		38	1				0.8	0.1		1		0.9	
Be		5							1				
Uniforn		4.5											0.9
[-]Hanc	Changi	0											
dynam	Changi	2.8										0.7	
Data/h		8	0.1					0.9					
Regule		9	0.1		0.3	0.3		0.1		0.1			

Figure 3 – Requirements x Failure Modes matrix

This gives a measure of the total requirements loss that each Failure Mode would cause if not mitigated against.

- The column of aggregate values displays, for each Requirement, the total expected loss of that Requirement due to the impact of Failure Modes. For Requirement R, this is computed as:

  A-priori-loss(R) = Weight(R) * (∑ (FM ∈ Failure Modes): Impact(FM,R) * Likelihood(FM))

  This gives a measure of the loss of each requirement due to all the (unmitigated) Failure Modes.

The tool provides bar-chart displays of these. Figure 4 shows the Failure Modes bar chart.

Note that it is possible for the aggregate loss computed for a requirement to exceed the original value of the requirement! For example, requirement number 1, "Allow issuing of help requests", was originally weighted at 10, and yet has an aggregate unmitigated loss computed to be 41. This is because there are multiple ways in which the requirement may be impacted. Indeed, two of them each lead to complete loss of that requirement should they occur. Nevertheless, we have found this to be a useful computed measure - it indicates just how much reduction of failure mode impacts remains to be accomplished by mitigations. In application to spacecraft mission assurance and planning, we have found that in practice people often employ sufficient mitigations to achieve some, often most, of a requirement, or recognize that a requirement is too expensive to achieve, and remove it entirely (i.e., decrease their ambitions). Removing requirements is more appropriate when using this approach for planning than for assessment.

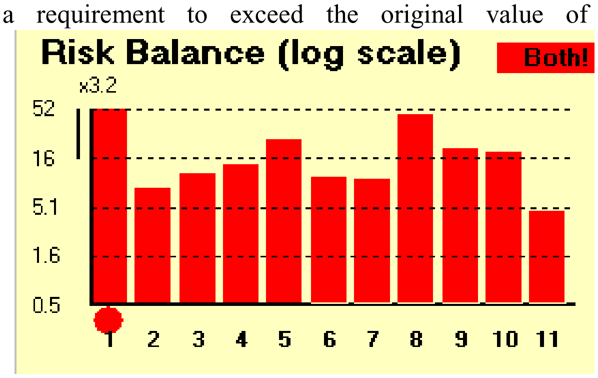

Figure 4 –Failure Mode's A-priori-loss

4. MITIGATIONS ELICITATION

The third step is the elicitation of Mitigations - the actions being taken to reduce the likelihood and/or impact of Failure Modes. For example, a design that included a backup power source at the TRMCS center would mitigate the "Power outage at center" Failure Mode. This step also includes the determination of how much each Mitigation reduces each Failure Mode.

Mitigations Elicitation - Methodology

For assessment purposes, mitigations will be found within the design, the implementation plan, etc. Personnel knowledgeable of the design details, implementation plan details, etc., will need to be involved in this step.

We have postulated 14 mitigations that our hypothetical TRCMS system employs – see Figure 5. For example, number 1, "Backup power source at center" suggest a fairly obvious approach to providing continuity of power. Like the Failure Modes, these Mitigations are very high-level. As the design progresses, an assessment at that stage would determine more detailed and design-specific failure modes and mitigations.

- 1:Backup power source at center
- 2:Encrypted data transmission
- 3:Passwords for access to data
- 4:Cross-check multiple monitors' readings
- 5:Triage plan
- 6:On-call services for peak demand
- 7:Physical replication of data
- 8:Standard communication protocols
- 9:Pre-planned responses
- 10:Alternative communication mechanisms
- 11:Distributed assistance centers
- 12:High connectivity to healthcare providers
- 13:Service tiers and options
- 14:Backup financial insurance

Figure 5 - Mitigations

Mitigations Elicitation – Tool Support

The DDP tool's support for representation and elicitation of Mitigations is similar to that for Requirements and Failure Modes. Mitigations do not have a weight or likelihood.

In a similar manner to the relationship between Failure Modes and Requirements, Mitigations can have different effects on different Failure Modes. The DDP tool maintains a Mitigation x Failure Mode matrix whose rows are Mitigations, and columns are Failure Modes. Each cell holds the effectiveness value of the cell's row's Mitigation on the cell's column's Failure Mode. An effectiveness value is a number in the range 0 to 1, where 0 corresponds to no effect whatsoever, and 1 corresponds to completely effective at mitigating the Failure Mode. An empty cell is equivalent to an entry of 0.

Figure 6 shows the effectiveness matrix for these Mitigations on the TRMCS Failure Modes. For example, the first row and column (shown highlighted) correspond to the Mitigation "Backup power source at center" and Failure Mode "Power outage at center". The inner cell holds the value 0.99, indicating that a backup power source will almost completely mitigate this Failure Mode. This is plausible, since there is a small chance that the backup power source itself might be inoperative when needed, but generally speaking will be sufficient. Of course, the determination of its sufficiency will require the judgment of appropriately skilled personnel, who understand the needs for, and capabilities of, backup power sources.

The tool automatically calculates some aggregate values for impacts *taking the current set of mitigations into account*. These are shown in the second row from the top, and third column from the left:

- The row of aggregate values displays, for each Failure Mode, the total expected risk of that Failure Mode *taking the current set of Mitigations into account*. For Failure Mode FM, this is computed as:

  Mitigated-Impact(FM) = A-Priori-Impact(FM) * (1 - ($\prod$ (M $\in$ Mitigations): (1 - Effect(M,FM)))

  This gives a measure of the total requirements loss that each Failure Mode would cause, taking mitigations into account.

- The column of aggregate values displays, for each Mitigation, the maximum expected risk savings application of that Mitigation would achieve. For Mitigation M, this is computed as:

  Mitigation(M) = ($\sum$ (FM $\in$ Failure Modes): A-Priori-Impact(FM) * Effect(M,FM))

  This gives a measure of the total benefit that each

PACTxFM Col = Power outage at center
Row = Backup power source at center

	FMs	Power	Insecu	Unauth	False	Overwh	Loss	New	Disast	Commu	Rudim	$
PACTs	FoM\R	0.1036	1.6	5.5	7	0.2625	0.0918	2	0.945	2	13.16	0.135
Backu	52.3	0.99					0.1					
Encryp	6.4		0.8									
Passw	5.5			0.5								
Cross-	7				0.5							
Triage	44					0.5			0.7			
On-call	22.5					0.9						
Physic	10.1						0.99					
Standa	8							0.8				
Pre-pl	13.5								0.3			
Alterna	18									0.9		
Distrib	54.58	0.8				0.3					0.3	
High	58					0.7			0.9			
Servic	4.05											0.9
Backu	3.15											0.7

Figure 6 – Mitigations x Failure Modes matrix

mitigation provides.

5. ASSESSMENT CALCULATION

Design assessment hinges on estimating how well the design mitigates the failure modes, and thereby meets the requirements.

Assessment Calculation – Tool Support

The DDP tool calculates the status of the impacts on Requirements by Failure Modes, taking into account the elicited information of Requirements, Failure Modes, Mitigations and their attributes and relationships. The tool makes available several visualizations of this information. For assessment purposes, the key such visualizations are the Requirements-centric view and the Failure-Modes-centric view.

Requirements-centric View of Outstanding Risk

Figure 7 shows the chart of the Requirements as impacted by all of the (completely unmitigated) Failure Modes. The red portion of the bars indicates loss of Requirements caused by Failure Modes, while the blue portion indicates Requirements that are unaffected by Failure Modes. It is normal for the bars to be mostly or totally red at this point, so the completely blue bar for Requirement 9.1 suggests that either it is a trivially satisfied requirement, or, more likely, that there are as-yet unidentified Failure Modes that would impact it.

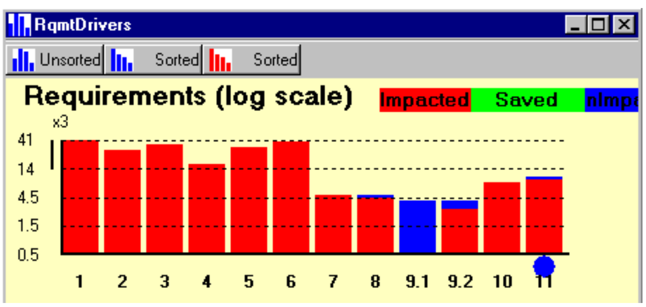

Figure 7 – chart of Requirements, unmitigated

Turning "on" all of our hypothesized mitigations gives the chart shown in Figure 8. Here, the green portions show the "savings" due to the Mitigations, and the red portions show the residual loss-of-Requirements despite the beneficial effect of the Mitigations. From this chart it is clear that there is still some significant loss of, especially, Requirements 1,3, 4 and 6. (Be aware that these are *log* scales. This is a heritage of our critical-systems setting, where we generally seek to push risk down to very low levels, for which a log scale is better suited.)

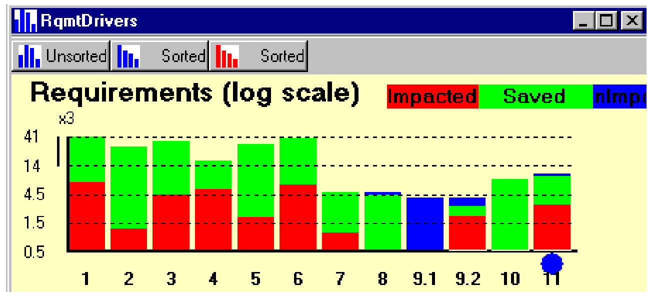

Figure 8 – chart of Requirements, fully mitigated

For a design that omitted the two security-related mitigations ("Encrypted data transmission" and "Passwords for access to data"), the Requirements chart would be that shown in Figure 9. Not surprisingly, Requirement 4 "Guarantee secrecy" is now the dominant problem area. Also, Requirement 11, "Regulations and Standards" has become more of a concern.

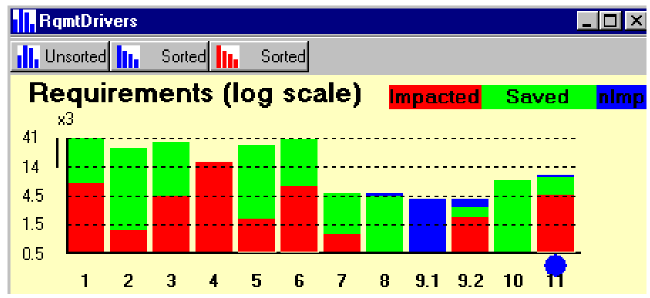

Figure 9 – chart of Requirements, partially mitigated

Failure-Modes-centric View of Outstanding Risk

Figure 10 shows the chart of the Failure Modes and the loss of requirements that they are causing, with all the Mitigations turned "on" (i.e., equivalent to Figure 8, but from the perspective of the Failure Modes).

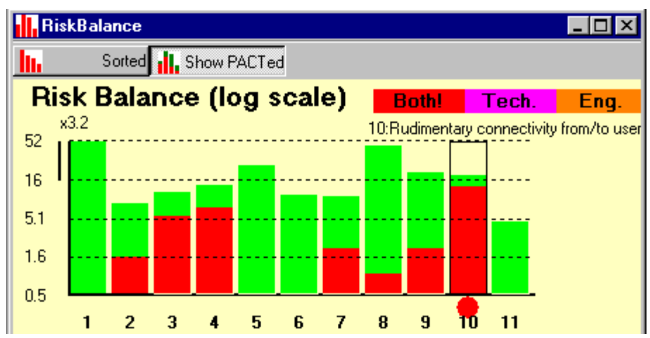

Figure 10 – chart of Failure Modes, Mitigated

From this chart it is clear that Failure Mode number 10, "Rudimentary connectivity from/to user" is the most problematic one for this design.

Assessment Calculation – Methodology

Accuracy of the calculations hinge upon the accuracy of the numerical quantities entered in the earlier stages. For this reason, the inclusion of experts whose combined knowledge spans the entire domain is strongly encouraged.

Even given such involvement, the methodology does not attempt to yield a single measure of adequacy (e.g., tempting though it would be to sum up the un-lost requirements, the tool does not do this). Rather, the methodology is aimed at identifying the relative strengths and weaknesses of a given design. This is a necessary step in assessing a design, and of considerable assistance to the assessment team.

6. DEVELOPMENT ASSESSMENT

The discussion and examples so far have illustrated the assessment of design. We believe a similar approach is applicable to the assessment of development, i.e., the process by which the design will be implemented.

We do not yet have realistic project experience to confirm this belief, so this is a working hypothesis. Within this section we describe the overall approach and status of our activities. Detailed examples are deferred to the appendix.

Development Failure Modes

Assessment of software development starts from a standard list of software development risks. The Software Engineering Institute (SEI) is one well-respected source of such information. In particular, the report Software Risk Evaluation Method [Sisti & Sujoe, 1994] presents a taxonomy of software risks. These have been encoded as development Failure Modes within the DDP tool.

Development Mitigations

SEI development practices serve as development Mitigations. For these, the SEI's Capability Maturity Model (CMM) for software [Paulk, et al, 1993] is used. Each of the five maturity levels (initial, repeatable, defined, managed, and optimizing) consists of several key process areas (KPA). For example, the KPAs of level 2 are requirements management, software project planning, software project tracking and oversight, software subcontract management, software quality assurance, and software configuration management. Each KPA is, in turn, supported by a few goals and is implemented by a group of activities. These activities have been encoded as the available set of Mitigations within the DDP tool.

Interestingly, we did not find any information as to *which* KPA activities address which risks, so we made our own estimate of this. Within the tool, we assigned a non-zero effectiveness value to every pair of KPA activity and software risk that we thought were related. At that time, we used the same non-zero effectiveness value throughout. [Feather et al, 1999] describes this encoding.

Tailoring through Inclusion of Quantitative Information

The aforementioned work established a qualitative framework for development assessment. For tailoring this to a specific assessment (e.g., of a development plan for a TRMCS design), *quantitative* information must be elicited and incorporated, in the following areas:

- Assigning assessment-specific effectiveness numbers to the Failure Mode x Mitigation pairs in their matrix. For example, consider the effect of Mitigation "Project commitments reviewed by senior management" on Failure Mode "Insufficient or unstable budget". If the development organization plans for recurring senior management budget reviews, then this will be very effective, and warrant an effectiveness measure of 0.9, say.
- Assigning impact values to the Failure Modes (SEI risks). In our experiments to date, we have simplified the DDP-based design assessment process. A single requirement serves as a placeholder for all concerns, and a loss-of-Requirements impact is assigned directly to each Failure Mode. For example, knowing that the TRMCS system will involve development of a critical communication component, development staff inexperience in this area might warrant a high impact measure.

7. CONCLUSIONS

Other work on assessment falls into two broad categories:

- High-level cost/schedule/risk assessment and management. E.g., the COCOMO work [Clark, B.; Devnani-Chulani, S.; Boehm, B., 1998]. Risk management tools are in use to gather and maintain risk status and tracking, but generally these tools employ comparatively simple means to assess the level of risk (e.g., ask an expert to qualitatively characterize a risk's likelihood and severity).
- Very detailed risk assessment. High assurance system engineering applies intensive assessment techniques, e.g., probabilistic risk assessment, to specific designs. E.g., the nuclear power industry uses these extensively [INSC].

Our approach fills the area in-between. We tailor assessments to modestly detailed levels of design and development information. The novelty of our approach hinges upon a quantitative approach that takes into account requirements, failure modes, and mitigations. This enables us to conduct assessments to both design and development plans. Our assessment calculations yield relative indications of which requirements are at risk, which Failure Modes are the most problematic, and which Mitigations are most critical.

ACKNOWLEDGEMENTS

The research described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not constitute or imply its endorsement by the United States Government or the Jet Propulsion Laboratory, California Institute of Technology.

REFERENCES

[Clark, B.; Devnani-Chulani, S.; Boehm, B., 1998] Calibrating the COCOMO II Post-Architecture model. Proceedings of the 1998 International Conference on Software Engineering, 1998, Page(s): 477 -480

[Cornford, 1998] S. Cornford. Managing Risk as a Resource using the Defect Detection and Prevention process. International Conference on Probabilistic Safety Assessment and Management, September 13-18, 1998.

[Feather et al, 1999] M.S. Feather, J.C. Kelly & J.D. Kiper, “Prototype Tool Support for SEI Process and Risk Knowledge”, *NASA 2nd Annual Workshop on Risk Management*, Morgantown, West Virginia, October 1999.

[Feather et al, 2000] M.S. Feather, S.L. Cornford & M. Gibbel. Scalable Mechanisms for Requirements Interaction Management, to appear in *Proceedings, 4th IEEE International Conference on Requirements Engineering*, June 2000.

[Paulk, et al, 1993] Mark C. Paulk, Bill Curtiss, Mary Beth Chrissis, Charles V. Weber. Capability Maturity Model for Software, Version 1.1. Technical Report CMU/SEI-93-TR-024, Software Engineering Institute, Carnegie Mellon University, February 1993.

[INSC] International Nuclear Societies Council ROLE OF RISK METHODS IN THE REGULATION OF NUCLEAR POWER Web Site at < http://133.205.9.136/~INSC/INSCAP/Risk.html>

[Sisti & Sujoe, 1994] F. Sisti and J. Sujoe. Software Risk Evaluation Method Version 1.0. Technical Report CMU/SEI-94-TR-019, Software Engineering Institute, Carnegie Mellon University, 1994.

APPENDIX – DEVELOPMENT ASSESSMENT

Elicitation

Figure 11 shows a portion of the Mitigation x Failure Mode matrix. The Mitigations are SEI CMM activities, and the Failure Modes SEI risks. The numerical values in the cells encode the effectiveness of those Mitigations against the Failure Modes. For example, the effect of the highlighted Mitigation "Project commitments reviewed by senior management" on highlighted Failure Mode "Insufficient or unstable budget" is set at 0.9. (If the development organization plans for recurring senior management budget reviews, then this will be very effective, and warrant such a high effectiveness measure.)

In a similar manner, quantitative measures of impact are assigned to each of the Failure Modes. For example, knowing that the TRMCS system will involve development of a critical communication component, development staff inexperience in this area might warrant a high impact measure of 0.7. See Figure 12.

Assessment

Once the quantitative information has been entered into the DDP tool, the same capabilities to calculate and display requirements loss can be employed for assessment purposes. Figure 13 shows the Failure-Modes-centric view of Requirements loss, given that all the Mitigations are "on". Failure Modes are shown in sorted (decreasing) order of Requirements loss, so there are many more, of lower impact, off the right of the image. The same kind of comparative assessment as was shown on design information can be performed to development information. For example, Figure 14 shows the same view when all of the SEI Software Quality Assurance activities have been

PACTxFM: 0 or empty = no effect; 1 = 100% effective

Hide UnChecked | Hide UnChecked | Value-Only Edit

PACTxFM Col = Insufficient or unstable budget
Row = Project commitments reviewed by senior management

		FMs	[-]Program Constraints									
		FMs	[-]Resources Risks				[-]Contract Risks			[-]Program Interface		
		FMs	Inadequ	Staff	Insuffici	Inadequ	Problem	Restricti	Program	Problem	Problem	F
PACTs	PACTs	FoM\RB	0.00014	0.06369	0.00231	0.01537	0.00000	0.00000	0.03203	0.4747	0.00000	0
[-]Requi Manage	SE group reviews requirements	42.2										
	SE group uses requirements as	4.4										
	Requirements changes are	40										
[-]Softw Project Planning	SE team participates on proposal	62	0.3									
	Software project planning	29	0.3									
	SE group participates in project	71.8	0.3									
	Project commitments reviewed by	74.6	0.6	0.1	0.9	0.6						
	Software life cycle is defined	23.3										
	Software development plan is	19.9										
	Software project plan	20										
	Software work products are	12.5	0.1									
	Size estimates are derived by	3.5	0.1									
	Effort and cost estimates are	7.4	0.1									
	Critical computer resource	24.9	0.1									
	Software schedule is derived by	2.2	0.1									
	Software risks are identified,	47.8	0.3	0.1	0.6	0.3	0.3	0.3	0.1			
	Plans for SE facilities and tools	16	0.3									
	Software planning data are	1.4	0.2									
[-]Softw Project Traking and	Software development plan used	13.8	0.3	0.1	0.3	0.1						
	Software development plan is	8.8										
	Software commitments and	9										
	Approved changes are	16.4										
	Size of work products are	18.2	0.1	0.1	0.1	0.1						

Figure 11

RxFM Col = Staff inexperiences, lacking knowledge or skills
Row = TRMCS software reliable

	FMs	[-]Program Constraints													[-]Dev
	FMs	[-]Resources Risks				[-]Contract Risks			[-]Program Interfaces Risks						[-]Dev
	FMs	Inadequ	Staff	Insuffici	Inadequ	Problem	Restricti	Program	Problem	Problem	Problem	Problem	Unrespo	Political	Impler
Rqmts	Totals	7	9	7	3	3	3	7	7	7	7	3	3	3	7
TRMCS	334	0.7	0.9	0.7	0.3	0.3	0.3	0.7	0.7	0.7	0.7	0.3	0.3	0.3	0.7

Figure 12

turned "off". Many of the Failure Modes bars have increased in height, indicating additional risk.

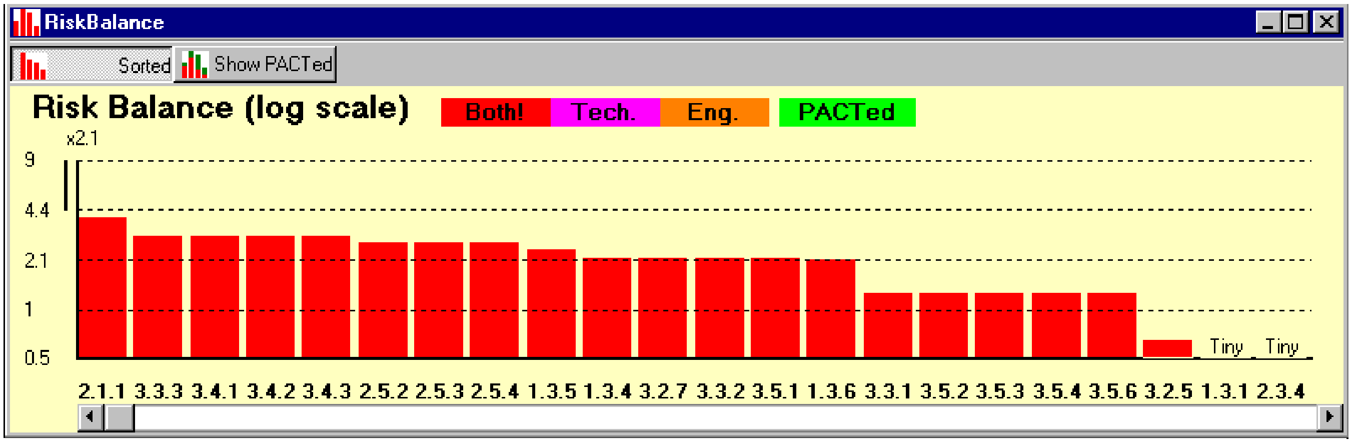

Figure 13 – sorted Failure Modes with all Mitigations active

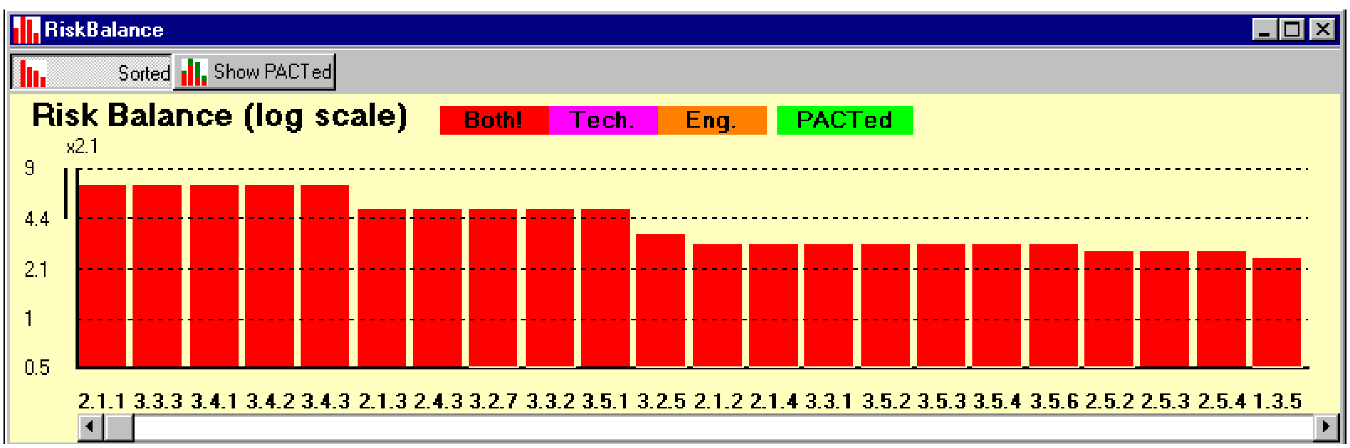

Figure 14 – sorted Failure Modes with all but Software Quality Assurance Mitigations active